A comparison of supervised learning algorithms

Bachelor Thesis

by

Ansar Aynetdinov

(577368)

Humboldt-Universität zu Berlin

School of Business and Economics

Ladislaus von Bortkiewicz Chair of Statistics

First Examiner: **Prof. Dr. Wolfgang Härdle**

Second Examiner: **Prof. Weining Wang**

Thesis Supervisor: Dr. rer. nat. Sigbert Klinke

In partial fulfillment of the requirements

for the degree of

Bachelor of Science (B.Sc.) in Economics

Berlin, August 19th 2018

Table of contents

List of Tables

List of Figures

1 Introduction

The fields of machine learning and computational statistics have gained a lot of attention in the last years, and every now and then buzzwords like “Artificial Intelligence” and “Predictive Models” get mentioned in news (e.g. The Economist, 2016). It is not surprising, since machine learning algorithms have become crucial tools for improving products of many companies in various industries. Especially nowadays, when the amounts of data available are larger than ever before, it is easier for analysts to gain new insights and to perform analyses that are not limited by the scarcity of data. Thus, it is a natural consequence that certain methods and algorithms, which were created before the universal increase in computational powers and digital upheaval in many industries, are used more than ever before.

However, while there are also a lot of the statistical and machine learning tools and methods used, it may be sometimes unclear which method should be used on a specific task. From my own experience working in an IT company with a strong analytical team, it can happen that personal preference of an analyst plays bigger role than a deliberated decision, when it comes to choosing the suitable method for a given task. As a matter of fact, it is often hard to say beforehand, whether the results of different methods applied will be different. But the consequences of applying the wrong tool may be incorrect future decision making and, potentially, a loss of potential revenue of a company. Therefore, it is an important topic, which affects many businesses in a direct way.

It made me think, if the use of different methods would really make a significant difference in results within the framework of, for example, a binary classification task? How strong are the differences? Is there a single best supervised learning algorithm?

That is why the focus of this thesis lies on the comparison of results that are generated by the supervised learning algorithms. Since there are numerous supervised learning algorithms available nowadays, that can solve a binary classification problem, this thesis concentrates only on several algorithms, that are most widely known.

Most of the big companies with a lot of capital develop their own learning algorithms, however it is very hard, if not impossible, to obtain those algorithms and compare them. They are also suited to specific needs of each company and thus make a comparison biased towards a certain task. Startups, in turn, may opt for the algorithms that are open to the public and that can be easily compared with each other.

It is important to note that the computational requirements and duration of the algorithms will not be the main focus of this thesis, despite the fact that they can play not the last role when choosing an algorithm in a business environment.

The binary classification task, that is going to be solved in the following analysis, is whether an American inhabitant earns more or less than 50 thousand dollars a year.

This thesis is organised as followed. Section two describes the dataset, which was used for the analysis, in detail. Section three contains the detailed description of algorithms that were chosen for the later analysis and comparison, and the section four describes the analysis and the results of it. Finally, section 5 summarises the thesis and attempts to give an answer to the questions stated above.

2 Data

For the purposes of this thesis, the American Community Survey (ACS) Public Use Microdata Sample (PUMS) of the year 2016 (one-year estimate) from the US Census Bureau was used. Each PUMS file represents a sample of actual responses to the ACS, which is an ongoing survey that provides vital information on a yearly basis about the American nation and its people, namely about jobs and occupations, educational attainment, veterans, whether people own or rent their homes, earnings and other topics. It was started in 2005 and replace once-a-decade census form that was decreasing overall census response rates and jeopardising the accuracy of the count (US Census Bureau, 2018).

ACS one-year estimates summarise responses received in a given year for all states and each record in the PUMS represents a single person. And since the 2016 PUMS contains data on approximately one percent of the United States population, it is a weighted sample that was drawn from the whole US population.

2.1. Data preparation

The initial PUMS file had more than three million observations and 284 variables. As mentioned in the introduction, the upcoming analysis will focus on earning of American inhabitants, hence it makes more sense to focus on those, who are able to earn money in the first place. The minimum age to work without any restrictions in every industry in the United States is 18 (U.S. Department of Labor, 2011), therefore individuals that are younger than 18 were removed from the data set. Furthermore, a lot of variables in dataset are of no interest for the further analysis either because there is a lot of data missing in them, or they provide some specific information about a more general variable, or there simply doesn't seem to be a connection between those variables and someone's ability to earn money. For example, variable "Veteran period of service" seems to show no connection to one's earning on a grand scale and variable "Interests, dividends, and net rental income past 12 months" is redundant, when there is a variable "Total person's income".

Besides, the variable "Total person's income" (PINCP), which is a target variable in the further analysis, was transformed from a numeric to a binary variable with categories "earned less than 50 thousand dollars (<50k)" and "earned at least 50 thousand dollars (>=50k)". The variable "Educational attainment" (SCHL) was also recoded into a variable "school" in order to reduce the number of classes and remove empty ones. The same applied to the variable „Marital status“ (MAR).

It means that the dataset, which will be analysed further, has ca. 2.5 million observations and ten variables. The full list and description of the customised variables used in the analysis can be found in Appendix.

The dataset was also split into three sets with a 60/20/20 split – a training set, a validation set and a test set respectively. The observations were drawn randomly, which ensure that the distribution of the data in each set of the data is equal to each other.

2.1.1. Missing values

Luckily, the dataset had almost no missing values, or the NAs had a specific meaning to them, which made it possible to put them in a separate category. For example, NAs in the variable „Class of worker“ meant that the person was not in labor force and it last worked five years ago or that it was never employed. Thus, the NAs were recoded as a separate category. The similar applied to the variable „Usual hours worked per week past 12 months“. There the missing values meant that the person did not work past twelve moths and thus were recoded as numeric „0“.

The only true missing values were in the variable „school“. In the initial full PUMS file, NAs in the variable „school“ meant that the person is less than three years old. After removing people younger than eighteen years old, this doesn’t explain the missing values anymore. There were only 70 thousand missing values, which makes up only about two percent of the whole sample size. Therefore, these cases were simply removed form the dataset, since such a small percentage is not significant for the further analysis.

3 Methods

3.1. Classification and Regression Trees (CART)

Classification and regression trees is a non-parametric decision tree learning algorithm that produces either a regression or a classification tree, depending on whether the dependent variable is categorical or numeric, respectively. It also can handle both continuous and categorical predictors. Since its first introduction by Breiman et al. (1984), CART appeals to many researchers and analysts due to its non-strict requirements and ease of interpretation.

CART algorithm is designed to represent decision rules in a form of binary trees, in which only only yes/no questions are asked, i.e each node has only two branches. Starting from the root node, all available variables and their possible values are looked through in order to find the split *s* – a combination of a variable from the available data and the appropriate question value. An optimal split *s** is the one that maximizes homogeneity (i.e. reduces variance) inside the two subsets that result from splitting. The process is iterated for each of the subsets until the tree reaches its optimal size. The bottom of the tree is represented by terminal nodes, which constitute the final decision rule *T**. (Andriyashin et al., 2008)

3.1.1. Construction of the classification tree

In the learning sample for a J class problem, let N_j be the number of observations in class j. The priori class probabilities can then be defined as proportion of the classes in the population:

$$\pi(j) = \frac{N_j}{N} \tag{3.1.1}$$

where $j = 1,...,J$.

Similarly, $N(t)$ is the number of observations in the node t, and $N_j(t)$ is the amount of class j observations in the node t. The probability that an observation of class j will fall into node t is given as:

$$p(j,t) = \pi(j)\frac{N_j(t)}{N_j} \tag{3.1.2}$$

From here it follows that $p(t) = \sum_j p(j,t)$. Now we can derive that the conditional probability that an observation falls into node *t*, given its class *j*, is:

$$p(j\,|\,t) = \frac{p(j,t)}{p(t)} = \frac{N_j(t)}{N(t)} \tag{3.1.3}$$

In other words $p(j,t)$ are the relative proportions of class *j* observations in node *t* (Breiman et al., 1984).

As mentioned above, the classification tree is built in accordance with a splitting rule - a rule that determines split $s*$ at each node. The goal is to create two more homogenous groups by splitting the initial less homogenous one (the parent node t_P) into two parts – left and right child nodes t_L and t_R. The split $s*$ contains one variable x_i from the matrix of explanatory variables X ($X = X_1, \ldots, X_p$ where p is the number of observations in the learning sample) and a question value $x*$. A question „Is $x_i < x*$?“ then splits the data. (Andriyashin et al., 2008)

Homogeneity in a node is calculated with help of the the impurity measure $i(t)$, which is determined by the impurity function $\phi(t)$:

$$i(t) = \phi[p(1\,|\,t), p(2\,|\,t), \ldots, p(J\,|\,t)] \tag{3.1.4}$$

Then the goodness of split *s* can be measured as a reduction in impurity at the node *t:*

$$\Delta i(s,t) = i(t) - p_R i(t_R) - p_L i(t_L) \tag{3.1.5}$$

where p_R and p_L are proportions of observations that fall into t_L and t_R respectively. (Breiman et al., 1984)

Therefore, at each following node the following problem is solved:

$$s* = \arg\max_s \Delta i(s,t) = \arg\max_s[-p_L i(t_L) - p_R i(t_R)] = \arg\min_s[p_L i(t_L) + p_R i(t_R)] \tag{3.1.6}$$

The maximum tree T_{MAX} is the tree containing the maximum number of nodes for a given data set. In other words, it is a tree built by applying equation (3.1.6) to the original data set and resulting split data portions until the following condition holds. This condition defines T_{MAX} as the tree where each terminal node contains only observations belonging to the same class *j*:

$$\forall t \in \tilde{T}\ \exists j : p(j\,|\,t) = 1 \tag{3.1.7}$$

where $\tilde{T}$ is the set of terminal nodes of a tree *T*.

The next important step is to define the impurity function $\phi(t)$. Many different impurity functions can be used, but two of the most commonly used for CART are Twoing and Gini splitting rules. According to Breiman (1996) all criteria should produce similar results, when the number of values of the dependent variable small. Since the dependent variable in the further analysis has only two values, any criteria could be chosen. Thus, Gini splitting rule was the rule of choice.

3.1.3. Gini splitting rule

Gini impurity is a measure of how often a randomly chosen observation from the set would be incorrectly classified, if it was randomly classified, according to the class distribution in the sample. It can be defined as:

$$i(t) = 1 - \sum_{j=1}^{J} p^2(j\,|\,t) \tag{3.1.8}$$

The splitting rule employing the Gini impurity (derivation from (3.1.5)) can be written down as:

$$\Delta i(t) = -\sum_{j=1}^{J} p^2(j\,|\,t) + p_L \sum_{j=1}^{J} p^2(j\,|\,t_L) + p_R \sum_{j=1}^{J} p^2(j\,|\,t_R) \tag{3.1.9}$$

and (3.1.6) then takes the following form (Andriyashin et al., 2008):

$$s* = \arg\max_{s}[-\sum_{j=1}^{J} p^2(j\,|\,t) + p_L \sum_{j=1}^{J} p^2(j\,|\,t_L) + p_R \sum_{j=1}^{J} p^2(j\,|\,t_R)] \tag{3.1.10}$$

3.1.3. Minimal cost-complexity pruning

Another important decision to make, when using classification trees, is to determine an optimal size of the tree. A maximum tree may be a good way to explain existing patterns in the available, but it is overfitted, when it comes to making predictions on an unknown set of data. On the other hand, a small tree may be not specific enough and also have a larger out-of-sample error, since it can’t classify precisely enough. One possible solution for this problem, proposed by Breiman et al. (1984) is cost-complexity pruning.

The cost-complexity measure of a tree is defined as:

$$R_\alpha(T) = R(T) + \alpha\,|\tilde{T}| \qquad (3.1.11)$$

where $T \leq T_{MAX}$ is any subtree of the maximum tree, R(T) is the misclassification cost, or the internal misclassification error of the tree, $|\tilde{T}|$ is the number of terminal nodes, or complexity of the subtree, and $\alpha \geq 0$ is the complexity parameter.

While α can take on an infinite number of values, there is only a finite number of subtrees of a maximums tree. Therefore, if there is an optimal tree for a given α, it will remain optimal, unless α will change.

The optimal subtree $T(\alpha)$ is defined by the conditions:

(i) $R_\alpha[T(\alpha)] = \min\limits_{T \leq T_{MAX}} R_\alpha(T)$

(ii) If $R_\alpha(T) = R_\alpha[T(\alpha)]$, then $T(\alpha) \leq T$

The result of this pruning is a sequence of subtrees $T_1 > T_2 > \ldots > \{t_1\}$, where $\{t_1\}$ is the root node, for which the sequence of alphas, $\{\alpha_k\}$, is increasing. By applying the method of V-fold cross-validation to the sequence of subtrees, an optimal tree can be determined.

However, selecting a tree with the minimum value of $R^{CV}(T)$ is not always the best choice, because usually the whole range of $R^{CV}(T)$ which satisfy $R^{CV}(T) < R^{CV}_{min}(T) + \varepsilon$ for small ε > 0. Besides, running a V-fold cross-validation procedure might give slightly different results. Thus, a so called one standard error rule can be applied, according to which a value of $\hat{R}^{CV}(T)$ within one standard error can be chosen (Andriyashin et al., 2008)

3.2. Multiple Logistic Regression

Multiple logistic regression represents one of the regression analysis methods. It describes the relationship between a binary or dichotomous variable Y and multiple explanatory variables denoted by X, which represents the whole set of independent variables $x_1, \ldots, x_k$. The dichotomous response variable is what makes the multiple logistic regression model different from the standard multiple regression model.

3.2.1. Introduction to Multiple Logistic Regression

In the logistic regression, the conditional mean (mean value of the outcome variable, given the values of the independent variables), expressed as $E(Y\,|\,X)$, can only be greater than or equal to zero and less than or equal to one (i.e. $0 \leq E(Y \Big| X) \leq 1$), compared to the multiple regression model, where the conditional mean ranges between $-\infty$ and $+\infty$. The reason for this is that a probability can't be greater than one or less than zero. Thus, the conditional mean follows an S-shaped distribution, which is given by the logistic function.

In order to simplify notation, the conditional mean $E(Y\,|X)$ is further denoted as $\pi(X)$, and it is calculated as:

$$\pi(X) = \frac{e^{\beta_0+\beta_1x_1+\ldots+\beta_kx_k}}{1 + e^{\beta_0+\beta_1x_1+\ldots+\beta_kx_k}} \tag{3.2.1}$$

Besides, the conditional distribution of the outcome variable given X is not normal with mean $E(Y\,|X)$, and the variance is not constant. In the case of dichotomous outcome variable, the outcome variable can be expressed as $Y = \pi(X) + \varepsilon$. Here the quantity ε may assume one of two possible values. If $Y = 1$ then $\varepsilon = 1 - \pi(X)$ with probability $\pi(X)$, and if $Y = 0$ then $\varepsilon = -\pi(X)$ with probability $1 - \pi(X)$. Thus, ε has a distribution with mean zero and variance equal to $\pi(X)\ [1 - \pi(X)]$. That is, the conditional distribution of the outcome variable follows a binomial distribution with probability given by the conditional mean, $\pi(X)$ (Hosmer et al., 2013).

Since the dependent variable in the logistic regression follows the Bernoulli distribution, the independent variables need to be linked to the Bernoulli distribution (or binomial distribution with n = 1 trials). In order to accomplish this, a link function called 'logit' is used.

The logit transformation of $\pi(X)$ is defined in terms of $\pi(X)$ as:

$$\text{Y}= g(X) = logit[\pi(X)] = ln[\frac{\pi(X)}{1 - \pi(X)}] = \beta_0 + \beta_1x_1 + \ldots + \beta_k + \varepsilon \tag{3.2.2}$$

Now the probability of an event occurring or not is converted into a continuous variable that is linear with respect to the explanatory variables. $ln[\frac{\pi(X)}{1 - \pi(X)}]$ is also called log-odds of an event occurring (or not occurring).

3.2.2. Fitting the Logistic Regression Model

The regression parameters in the logistic regression are estimated with help of the maximum likelihood method. In general sense, it yields values for the unknown parameters that maximise the probability of obtaining the observed set of data. In order to apply this method, a function that expresses probability of the observed data as a function of unknown parameters (likelihood function), is needed in the first place. Afterwards, maximum likelihood estimators (MLE) of the parameters need to be calculated. They maximise the likelihood function, hence they agree most closely with the observed data (Hosmer et al., 2013). Here is how those values are found:

The conditional probability of $Y=1$, given X is denoted as $P(Y=1|X)=\pi(x)$, otherwise $P(Y=0|X)=1-\pi(X)$. Therefore, for the pairs (x_i, y_i) where x_i is the value of the independent variable and y_i is the value of the dependent variable for the i-th observation, the contribution to the likelihood function, when $y_i=1$ and $y_i=0$ are $\pi(x_i)$ and $1-\pi(x_i)$ respectively. Therefore, the contribution of the pair (x_i, y_i) to the likelihood function can be calculated as:

$$\pi(x_i)^{y_i}[1-\pi(x_i)^{1-y_i}] \tag{3.2.3}$$

As the observations are assumed to be independent, the likelihood function is given as:

$$l(\beta)=\prod_{i=1}^{n}\pi(x_i)^{y_i}[1-\pi(x_i)^{1-y_i}] \tag{3.2.4}$$

where $\boldsymbol{\beta}$ is a vector of parameters β_0, $\beta_1, \ldots,$ β_k and n is the number of observations.

Since it is easier to find values that maximize $\boldsymbol{\beta}$ with the log of the equation (3.2.4), the log-likelihood is defined as:

$$L(\beta)=\ln[l(\beta)]=\sum_{i=1}^{n}\left\{y_i\ln[\pi(x_i)]+(1-y_i)\ln[1-\pi(x_i)]\right\} \tag{3.2.5}$$

Values of $\boldsymbol{\beta}$ that are a result of solving the maximization problem in the equation 3.2.5 are the MLEs.

3.2.3. Interpretation of the regression coefficients

The regression coefficients in the logistic regression have also a different interpretation compared to the linear regression, because of the logit link function. In case of the categorical independent variable, $k-1$ dummy variables need to be created for $k>2$ categories. Then, a regression coefficient represents the natural logarithm of the ratio of the odds that Y=1 among observations of a certain category to the odds of Y=1 among observations of the left out, or reference, category:

$$OR = \frac{\frac{\pi(k_1)}{1-\pi(k_1)}}{\frac{\pi(k_{ref})}{1-\pi(k_{ref})}} = \frac{\frac{e^{\beta_0+\beta_{k_1}}}{1+e^{\beta_0+\beta_{k_1}}}}{\frac{1}{1+e^{\beta_0+\beta_{k_1}}}} = e^{\beta_{k_1}} \tag{3.2.6}$$

where k_{ref} is the reference category, and k_1 is one of the categories (Hosmer et al., 2013).

A continuous dependent variable, on the other hand, shows the difference in the log odds, when the continuous dependent variable increases by one unit.

3.3. Naive Bayes classifier

Naive Bayes classifier is a simple but very practical and useful Bayesian learning method. Its approach to classifying data is to assign the most probable target value, v_{MAP} (MAP = Maximum A Posteriori), given the attribute values $(a_1, a_2, \ldots, a_n)$ that describe the instance.

$$v_{MAP} = \arg\max_{v_j \in V} P(v_j \mid a_1, a_2, \ldots, a_n) \tag{3.3.1}$$

Applying the Bayes theorem to this expression gives us:

$$v_{MAP} = \arg\max_{v_j \in V} \frac{P(a_1, a_2, \ldots, a_n \mid v_j)P(v_j)}{P(a_1, a_2, \ldots, a_n)} = \arg\max_{v_j \in V} P(a_1, a_2, \ldots, a_n \mid v_j)P(v_j) \tag{3.3.2}$$

However, estimating various $P(a_1, a_2, \ldots, a_n \mid v_j)$ terms is complicated as the dimensionality increases. Therefore, the naive Bayes classifier utilises one simplifying assumption. It assumes that the attribute values a_i are conditionally independent, given the target value. So given the target value v_j, the values of different attributes do not affect each other, i.e. the probability of observing the conjunction $a_1, a_2, \ldots, a_n$ is just the product of the probabilities for the individual attributes, which gives us:

$$v_{NB} = \arg\max_{v_j \in V} P(v_j) \prod_i P(a_i \mid v_j) \tag{3.3.3}$$

where v_{NB} is the target value predicted by the naive Bayes classifier. Whenever the assumption of conditional independence is satisfied, v_{NB} is identical to v_{MAP} (Mitchell, 1997).

3.4. Feedforward neural network

While nowadays there are many different types of artificial neural networks (ANNs), this thesis focuses only on the simplest type – the feedforward neural network.

Simply put, the concept of ANNs was inspired by the basic fact that the biological learning systems in the animal and human brains consist of interconnected neural webs. ANNs remotely resemble them, in that they're built out of an interconnected set of nodes, or artificial neurons, which each neuron can produce some kind of an output from a given input. This output can be then transmitted to the next layer of neurons, and so on.

The feedforward neural network has a structure of an acyclic graph, meaning that there are no cycles or loops. When data is „fed" to the network, it moves only in one direction, forward, from the input node to the output nodes. Meanwhile, the network can have either none or multiple hidden layers between the input and the output layers. If a network has no hidden layers, it is called „perceptron". This type of networks, however, is limited in its usage, since it utilises a step activation function (a function that defines the output of a node, given the input:

$$o(x_1,\ldots,x_n) = \begin{cases} 1 & \text{if } \omega_1 x_1 + \ldots + \omega_n x_n > 0 \\ -1 & \text{otherwise} \end{cases} \tag{3.4.1}$$

Where $o(x_1,\ldots,x_n)$ is the output, x_i is an input and ω_i is a weight that determines the contribution of x_i to the output.

It follows that while the perceptron can represent many primitive boolean functions like AND and OR, it fails to represent such functions like XOR (exclusive OR). In other words, the perceptron fails to learn not linearly separable patterns, i.e. that can not be correctly classified with a straight line (Mitchell, 1997). Therefore, using the multilayer perceptron (MLP), that has at least one hidden layer, is advised, in order to address possible non-linearities in the data.

3.4.1. Multilayer networks and the backpropagation algorithm

Multilayer networks are able to depict nonlinear nonlinear patterns, compared to the perceptron. The weights for a multilayer network are learned by the backpropagation algorithm. It utilises gradient descent for minimisation of the error between the network output values and the target values from training data for these outputs. Gradient descent looks through the hypothesis space of possible weight vectors to find the weights that best fit the training examples by the training error (Mitchell, 1997). The training error for a multilayer classification network is defined as:

$$E_C = \sum_{i=1} \sum_{j=1}^{C} y_{ij} log[\sum_{j=1}^{N} \omega_j \Lambda \omega_{i0} + \sum_{i=1}^{p} \omega_{ij} x_i)] \tag{3.4.2}$$

with C representing the amount of classes, ω_{ij} denoting weight of the input from node i to hidden unit j, ω_j denoting weight of the connection between a hidden and an output unit, Λ being an activation function, $y_{ij} = 1$ if observation i belongs to class j and otherwise $y_{ij} = 0$.
The essence of the backpropagation algorithm can be captured as:

$$\omega_{ij,t} = \omega_{ij,t-1} - \lambda_t \frac{\partial E}{\partial \omega_{ij}} \tag{3.4.3}$$

where t is the time parameter that separates different steps of the gradient descent search, λ_t is a positive constant called learning rate with $lim_{t \to \infty} \lambda_t = 0$, ω_{ij} denotes weight of the input from node i to hidden or output unit j. The role of the learning rate is to moderate the degree to which weights are changed at each step of gradient descent.

Sometimes the weights are made to decay to avoid overfitting. Each weight is decreased by some factor during each iteration of the backpropagation algorithm, thus large weights in the network are penalised. Smaller weights smooth out the activation function and makes it more linear. Weight decay can be formally captured as:

$$E' = E + w(\sum \omega_j^2 + \sum \omega_{ij}^2) \tag{3.4.2}$$

where $w \in [0,1]$ is the decay parameter (Klinke, 2018).

3.4.2. Universal approximation theorem

As Cybenko (1989) has shown, any continuous function can be approximated by a neural network that has only one hidden layer, given a sufficient amount of the nodes in this hidden layer. However, it is important to note that while a single hidden layer is sufficient, it may not be efficient, since the amount of nodes in the hidden layer, that can be required to yield some approximations may be quite high.

According to Hornik (1991), this universal approximation property of the feedforward neural networks is achieved because of the feedforward property of the networks and not the choice of the activation function. Therefore, any non-constant, bounded, and monotonically increasing continuous activation function can be chosen.

3.5. Performance evaluation metrics

3.5.1. Confusion matrix

In order to compare different classification methods, some kind of a performance metric is needed. Since this analysis deals with a comparison of binary classifiers, metrics that are derived from confusion matrix can be helpful. The confusion matrix represents four possible outcomes of a binary classification task: True Positives, False Positives, True Negatives and False Negatives, as shown in Table 3.1:

		True class	
		False	True
Predicted class	False	True Negatives (TN)	False Negatives (FN)
	True	False Positive (FP)	True Positives (TP)

Table 3.1: Confusion Matrix

The true positives and the true negatives are observations, that are correctly classified by the classifier, whereas false positives and false negatives represent type I and type II errors respectively. In our classification task false positives would include people, who are classified as earning more than 50 thousand dollars per year, but are actually earning less than 50 thousand dollars. False negatives, on the other hand, are people, who are classified as earning less than 50 thousand dollars a year, but are actually learning more than 50 thousand dollars.

The share of true positives and negatives in the whole set of predictions is defined as accuracy:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.5.1}$$

Another metrics are recall (True Positive Rate):

$$recall = \frac{TP}{TP + FP} \tag{3.5.2}$$

and specificity (True Negative Rate):

$$specificity = \frac{TN}{TN + FN} \tag{3.5.3}$$

While these metrics and other metrics similar to them can be used as tools to compare performance of classification models, they are easily influenced by class skewness (Fawcett, 2006). It means that they can show better performance of a model when the classes are skewed, compared to a case with balanced classes. For example, if a test dataset has 90% of observations belonging to one class, and some classifier predicts that simply observations belong to that class, this classifier would still have high accuracy. Whereas if the classes were balanced, accuracy of this classifier would have been far worse.

3.5.2. ROC analysis and AUC

In such cases receiving operator characteristics (ROC) analysis is very helpful. ROC graphs are able to provide richer a richer measure of classification performance than scalar measures like accuracy, since they decouple classifier performance from class skew and error costs (Fawcett, 2006).

ROC graphs are two-dimensional graphs where the y-axis depicts true positive rate (precision) and the x-axis depicts false positive rate (1 - specificity). Alternatively, the x-axis can also represent specificity with values condescending from one to zero. The ROC space is shown in Figure 3.1. If an ROC curve reaches point (0,1) in Fig. 1, it represents the perfect classifier. Thus, the more northwest an ROC curve reaches, the better the classifier is. The diagonal line illustrates random performance. Basically, such classifier predicts correct values 50% of the time. Any classifiers mapped on the right from the diagonal lines are showing worse than random performance, which makes them bad classifiers.

Area under the ROC curve (AUC) summarises the ROC performance in a single scalar, which allows for a simpler comparison of classification models. AUC is always between 0 and 1. The value of 0.5 represents the area under the diagonal line in the ROC space, thus making . Besides, AUC has an important statistical interpretation: it is equivalent to the probability that

the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance, assuming that "positive" ranks higher than „negative“ (Fawcett, 2006).
Because of its benefits and ease of interpretation, the AUC measure will be used in the further comparison of binary classification.

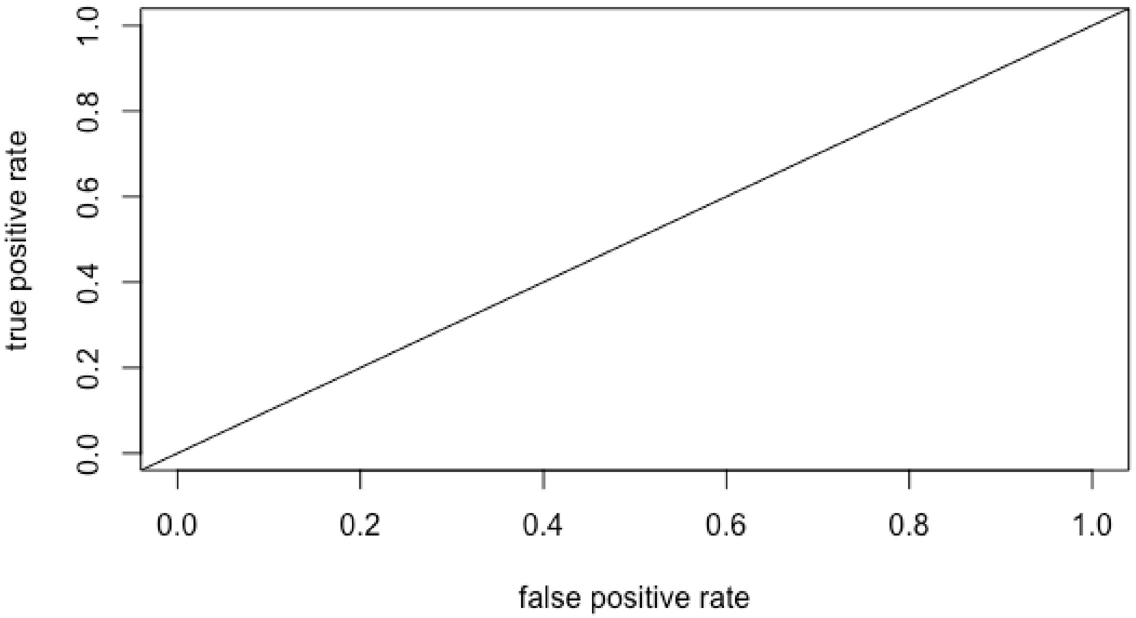

Figure 3.1: ROC space

4 Analysis

The first supervised learning algorithm that was applied to solve the binary classification problem, stated in the beginning, was the CART algorithm.

4.1. CART

4.1.1. Constructing the classification tree

Firstly, a maximum tree was grown, using the data from the training set. The result is a huge tree with more than thirteen thousand terminal nodes, where all the variables were used. After applying cost-complexity pruning with ten-fold cross-validation, the tree size reduced considerably, but it still remained quite large with 439 terminal nodes. Therefore, the one standard error rule was applied, in order to further reduce the size of the classification tree. The amount of terminal nodes now reduced down to 303. However, as one can see in the Figure 4.1, there is not much improvement happening, associated with the misclassification error, when the size of the tree increases after a certain point.

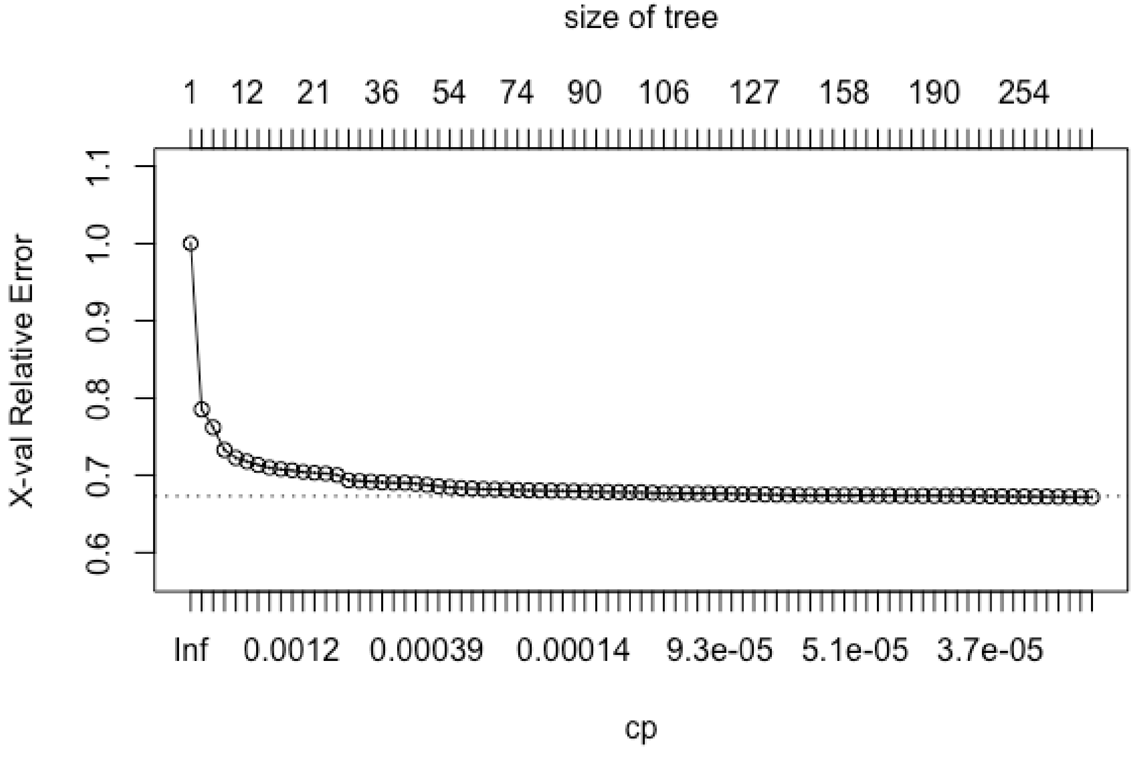

Figure 4.1: Visual representation of the relationship between the pruned tree size, the corresponding complexity parameters and the cross-validated misclassification error.

Moreover, many of the terminal nodes have 50 to 300 observations, which is negligible, taking into account that the training set contains almost 1.5 millions of observations.

It was also interesting to see, if the default rpart procedure in R would deliver different results, therefore a decision tree with default pruning rule was also computed. The default rpart rule is that any split that does not decrease the internal misclassification error by a factor of the complexity parameter, will not be added to the tree.

The results are indeed different, since the tree became considerably smaller and it uses only three variables out of twelve – Age, Hours worked per week and Educational attainment. Besides, it was possible to plot the tree itself (Figure 4.2) so that it would be descriptive enough for a reader. It makes use of one of the main advantages of the decision trees, i.e. ease of interpretability.

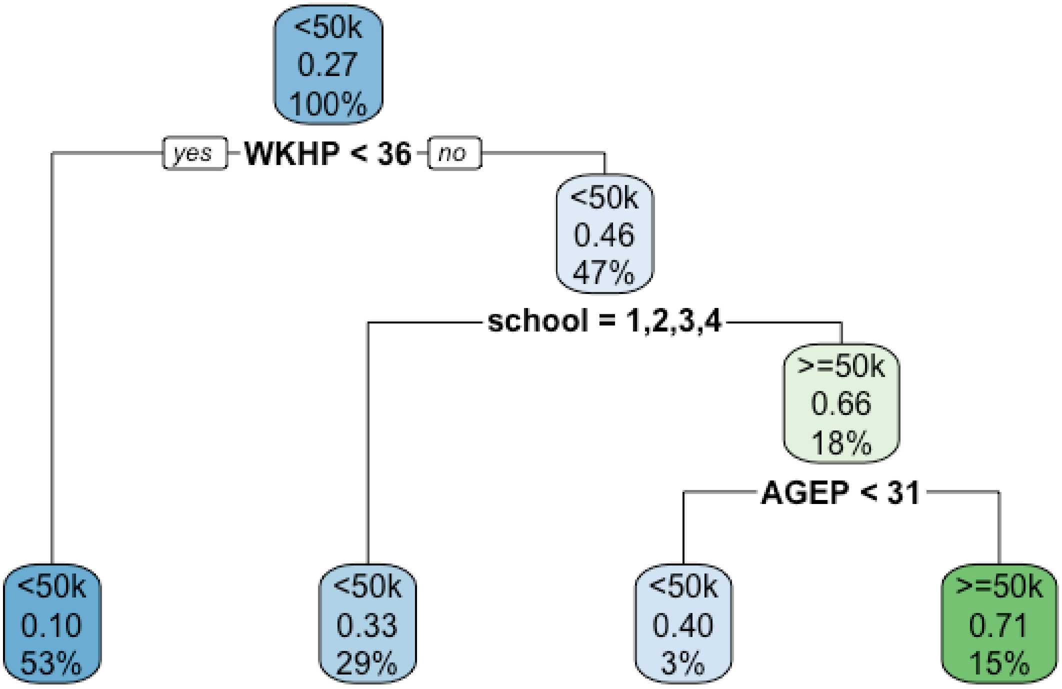

Figure 4.2: Classification tree based on the default pruning rule from rpart command in R

There are no surprising results in the decision rules. It is intuitive that if one works more hours, has a better education and is older (because it is often correlated with working experience), one will probably earn more money.

Next step is to see which tree predicts the annual income more accurate and choose one of them for the further comparison.

4.1.2. Tree selection

As described before, in order to select a tree for a further comparison with other algorithms, the AUC scores were compared. The performance of the trees was tested on a previously unseen validation set of the data. This ensures that the out-of-sample error is taken into account. The results are shown in the Table 4.1:

Tree version	AUC
Maximum tree	0.74
Cost-complexity pruned tree	0.75
One standard error rule pruned tree	0.75
R default pruned tree	0.7

Table 4.1: AUC measures of different classification tree versions

The results show that it would be best to choose the cost-complexity pruned tree with one standard error rule applied to it, since it has the same accuracy as the one without the one standard error rule, but is less complex.

It is worth to note that the difference between the AUC of this tree and the one that was pruned with default R pruning is 0.05, while the difference in the tree size is enormous. The R default pruned tree provides the best AUC-size ratio by far, and it is way easier to interpret. However, since the emphasis of this thesis lies on the comparison of prediction power of different algorithms and not on the interpretability, only the cost-complexity pruned tree will be considered in the further comparison.

4.2. Logistic regression

In order to build the logistic regression model, the independent variables needed to be firstly checked for multicollinearity to ensure stability and reliability of the estimated regression coefficients. In case of numeric variables, the usual Pearson correlation coefficient was used, and the association between the categorical variables was measured with help of the Cramer's V. The results are shown in Table 4.2:

	AGEP	COW	MAR	school	SEX	WKHP	DIS	RACWHT	WAOB
AGEP	1					-0.37			
COW		1	0.12	0.12	0.17		0.36	0.07	0.03
MAR		0.12	1	0.17	0.04		0.10	0.11	0.08
school		0.12	0.17	1	0.07		0.18	0.09	0.08
SEX		0.17	0.04	0.07	1		0.01	0.01	0.02
WKHP	-0.37					1			
DIS		0.36	0.10	0.18	0.01		1	0.02	0.08
RACWHT		0.07	0.11	0.09	0.01		0.02	1	0.40
WAOB		0.03	0.08	0.08	0.02		0.08	0.40	1

Table 4.2: Correlation matrix

Fortunately, almost all of the independent variables have weak or very weak correlations between them. Therefore, there should be no reason for multicollinearity in the logistic regression model, which would have drastically affected the regression coefficients.

Another preparatory step was to create $k - 1$ dummy variables from nominal variables with $k > 2$ categories, in order to be able to interpret the regression coefficients for these variables.

4.2.1. Fitting the logistic regression

The results of the initial logistic regression are shown in Table 4.3. The dependent dummy variable is „Earns a person more than 50 thousand dollars per year?“. Value of 0 means no and 1 – yes. A number near the variable name indicates the category in the initial, non-dummy variable.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-4.12	0.08	-54.90	< 0.001
school_1	-3.00	0.02	-140.59	< 0.001
school_2	-2.45	0.02	-134.01	< 0.001
school_3	-1.93	0.02	-106.06	< 0.001
school_4	-1.57	0.02	-82.91	< 0.001
school_5	-0.87	0.02	-48.54	< 0.001
school_6	-0.29	0.02	-15.52	< 0.001
school_7	-0.08	0.02	-3.52	< 0.001
SEX_1	0.75	0.005	156.06	< 0.001
AGEP	0.04	< 0.001	241.37	< 0.001

COW_1	0.18	0.01	17.38	< 0.001
COW_2	-0.02	0.01	-1.28	0.20
COW_3	0.22	0.01	17.21	< 0.001
COW_4	0.13	0.01	9.31	< 0.001
COW_5	0.93	0.02	58.08	< 0.001
COW_6	-0.55	0.01	-40.13	< 0.001
COW_7	0.05	0.02	3.24	0.001
COW_8	-1.11	0.06	-18.34	< 0.001
COW_9	-1.39	0.13	-10.85	< 0.001
MAR_1	0.41	0.005	82.81	< 0.001
DIS_1	-0.41	0.01	-51.35	< 0.001
WKHP	0.06	< 0.001	312.65	< 0.001
RACWHT_1	0.24	0.01	34.30	< 0.001
WAOB_1	-0.08	0.07	-1.07	0.29
WAOB_2	-0.34	0.08	-4.22	< 0.001
WAOB_3	-0.53	0.07	-7.33	< 0.001
WAOB_4	-0.14	0.07	-1.89	0.06
WAOB_5	-0.11	0.07	-1.48	0.14
WAOB_6	-0.51	0.08	-6.61	< 0.001
WAOB_7	0.13	0.08	1.67	0.10
Null deviance	1,693,150	on 1,456,869 df		
Residual deviance	1,170,426	on 1,456,840 df		
AIC	1,170,486			

Table 4.3: Results of the initial logistic regression

There are five insignificant variables in the initial logistic regression, which are marked grey. Insignificance means that the odds of Y=1 given the insignificant class is the same as of Y=1 given the reference class.

One possible solution for deciding which insignificant variables should be taken out of the model is a stepwise backward regression. In each iteration it deletes the variable with smallest partial correlation until the model becomes better (Klinke, 2018). The results of the stepwise regression are shown in Table 4.4.

	Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-4.20	0.02	-187.12	< 0.001
school_1	-3.00	0.02	-140.74	< 0.001
school_2	-2.44	0.02	-134.15	< 0.001
school_3	-1.93	0.02	-106.12	< 0.001
school_4	-1.57	0.02	-82.92	< 0.001
school_5	-0.87	0.02	-48.53	< 0.001
school_6	-0.29	0.02	-15.51	< 0.001
school_7	-0.08	0.02	-3.51	< 0.001
SEX_1	0.75	0.005	156.89	< 0.001
AGEP	0.04	< 0.001	244.47	< 0.001
COW_1	0.18	0.01	25.38	< 0.001
COW_3	0.23	0.01	21.35	< 0.001
COW_4	0.14	0.01	11.42	< 0.001
COW_5	0.94	0.01	65.10	< 0.001
COW_6	-0.54	0.01	-45.05	< 0.001
COW_7	0.06	0.01	4.25	< 0.001
COW_8	-1.10	0.06	-18.31	< 0.001
COW_9	-1.39	0.13	-10.81	< 0.001
MAR_1	0.41	0.00	82.83	< 0.001
DIS_1	-0.41	0.008	-51.44	< 0.001
WKHP	0.06	0.00	347.94	< 0.001
RACWHT_1	0.24	0.01	34.29	< 0.001
WAOB_2	-0.26	0.04	-7.39	< 0.001
WAOB_3	-0.46	0.01	-38.67	< 0.001
WAOB_4	-0.06	0.01	-5.04	< 0.001
WAOB_5	-0.03	0.02	-2.09	0.04
WAOB_6	-0.44	0.03	-14.30	< 0.001
WAOB_7	0.21	0.03	6.12	< 0.001
Null deviance	1,693,150	on 1,456,869 df		
Residual deviance	1,170,426	on 1,456,842 df		
AIC	1,170,484			

Table 4.4: Results of the stepwise backward logistic regression model

Now all the regression coefficients are significant on the significance level of $\alpha = 5\%$. The regression coefficients also represent the expected tendencies, when it comes to the expected income. Each additional year of life and each additional working hour per week increases the odds of earning more than 50 thousand dollars per year by approximately four and six percent respectively. Furthermore, being disabled or not being in a marriage having a decreases the same odds. Moreover, the widely discussed facts that being a white male gives an advantage, when it comes to yearly earnings, are reflected in this model as well.

One interesting observation from this model is that being self-employed in own not incorporated business (COW_6) gives worse odds of earning more than 50 thousand dollars per year than being not in labor force (COW_10, reference class).

4.2.2. Model selection

In order to see if the prediction power of the logistic regression was affected, the AUC scores of both models were compared on the validation set (Table 4.5). Both models have almost identical ROC curves and AUC scores, thus the stepwise backward logistic regression is preferred because of the lesser complexity of the model.

Logistic Regression Model	AUC
Initial model	0.86
Stepwise backward model	0.86

Table 4.5: AUC scores of the two logistic regression models

Since the logistic regression does not make discrete predictions like CART does, ROC curves for both logistic regression models look smooth (Figure 4.3) and consequently have bigger AUC scores than the CART models. This occurs because the output of the logistic regression predicts probabilities of Y = 1 $(Y = \pi(X))$, and ROC curve shows the relationship between true and false positive rates for each classification threshold. It also means that predicting actual classes requires a threshold for classification to be defined. It is a common practice to choose a threshold of 0.5, meaning that if $\pi(X)$ > 0.5, Y = 1, and Y = 0 otherwise. In our case, Y = 1 means that a person earns more than 50 thousand dollars per year and Y = 0 means that a person earns less than 50 thousand dollars.

However, since the classes in our dataset are not balanced and the ratio between the classes is 75/25, a threshold of 0.5 might be not suitable. Therefore, other threshold values were also tested in order to see, whether their performance would be better. Table 4.6 shows that threshold of 0.25 has the best performance, so it will be used in the following comparison. Figure 4.4 tells us that a smaller threshold allows for better sensitivity, on the cost of worse specificity of the classifier.

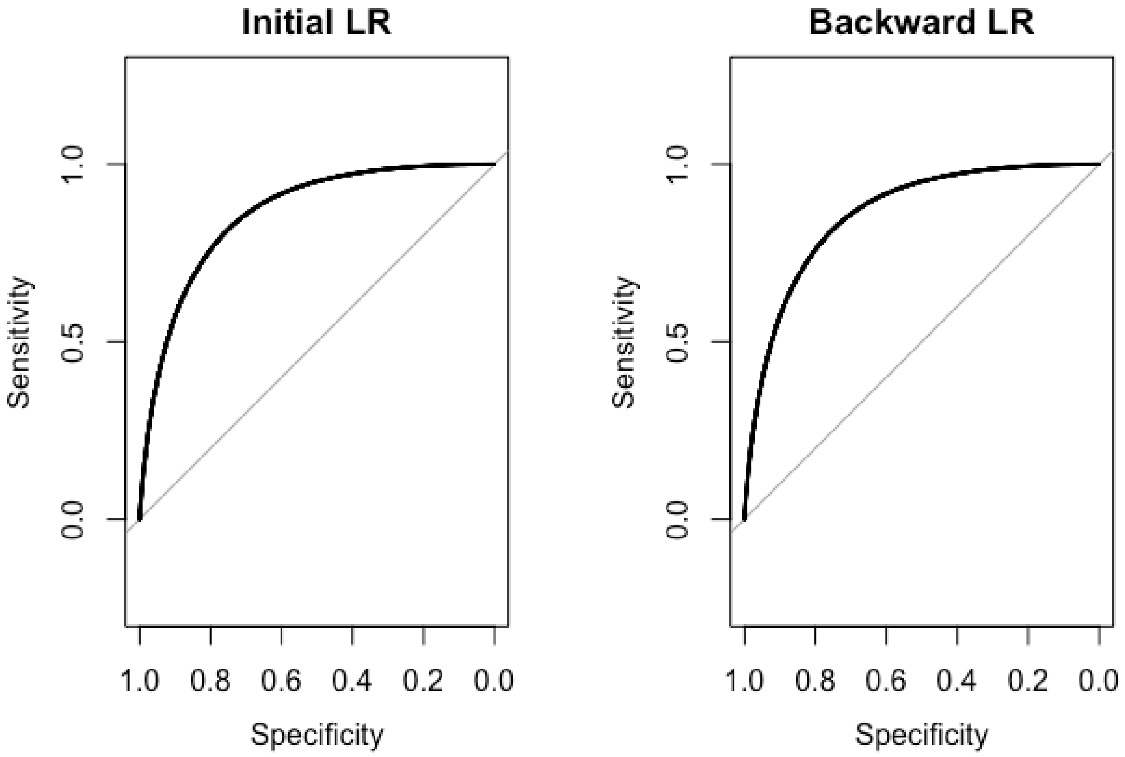

Figure 4.3: ROC curves of the two logistic regression models

Threshold	AUC
0.75	0.60
0.6	0.68
0.5	0.72
0.4	0.76
0.25	0.783
0.2	0.777

Table 4.6 : Logistic regression thresholds

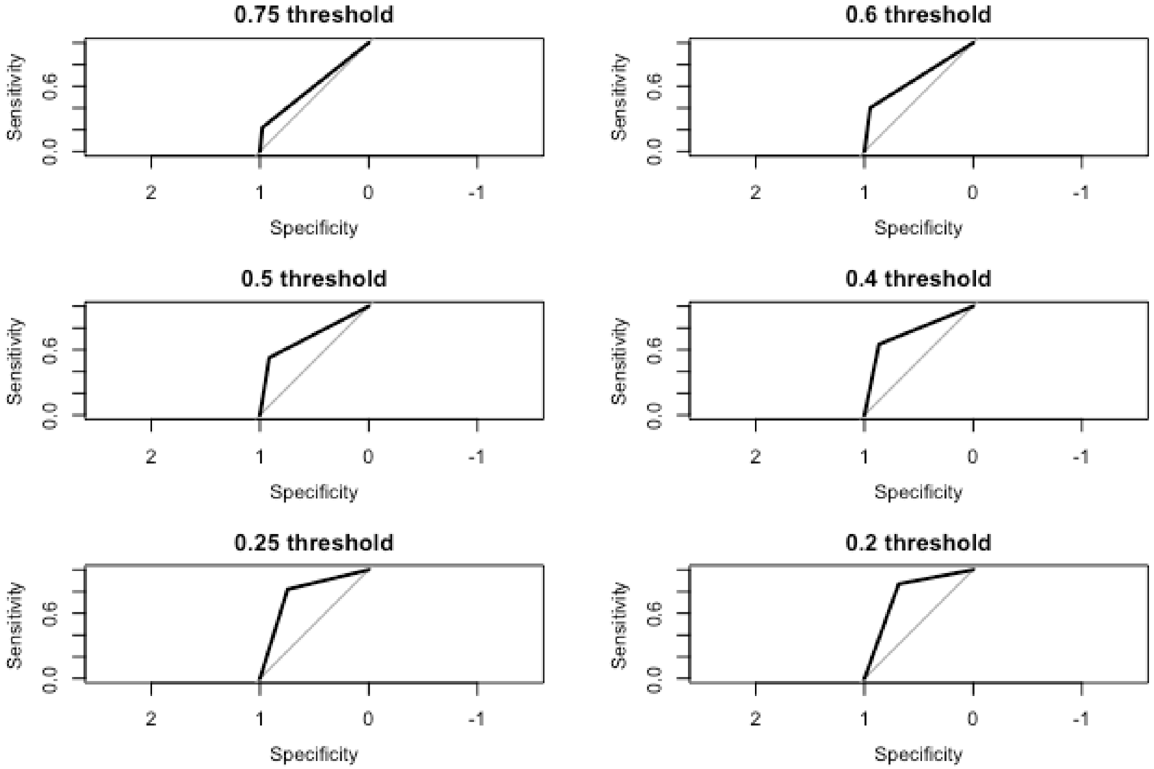

Figure 4.4: ROC curves of the stepwise logistic regression model with different thresholds

4.3. Naive Bayes classifier

Naive Bayes classifier usually makes an assumption about continuous variables, that they follow a normal distribution, given the target class label (Ng and Jordan, 2001). In our dataset the numeric variables are definitely not normal distributed (Figure 4.5), however, they are not continuous, but discrete. Therefore, Naive Bayes can handle these discrete values as separate classes, which is one solution for not normal distributed continuous variables.

However, there are two another options – either to use kernel density estimation to estimate the class-conditional distributions, or recode the numeric variables into categorical variables. To be safe, all methods were tested and compared to each other.

The variable „Age“ was split into four categories, that attempt to resemble both life and working experience stages:

1) 18-30 years old – young adults, who just or recently entered the workforce and whose main focus is to gain experience and knowledge

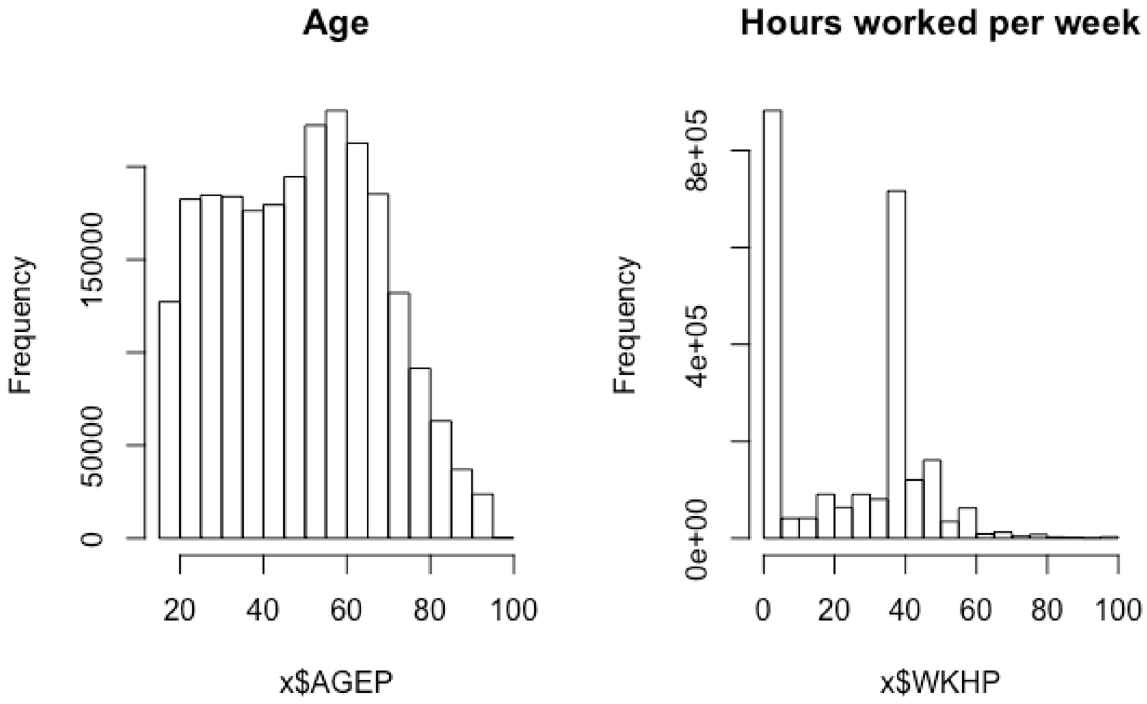

Figure 4.5: Histograms of the two numeric variables in the PUMS dataset

2) 30-45 years old – middle-aged people, who can call themselves professionals and start to get higher positions
3) 45-60 years old – middle-aged, well-experienced workers
4) 60 years and older - senior people, mostly retired

The variable „Usual hours worked per week past 12 months" was also split into four categories:

1) 0 hours – did not work
2) 1-34 hours – part-time workers
3) 35-59 hours – usual working hours, full-time jobs
4) 60 hours and more – people who are working a lot of additional hours

All versions of the Naive Bayes classifier were then trained on the training set and their AUC scores were compared based on the validations set. The results are listed in the Table 4.7 and shown in Figure 4.6. As one can see, the version with kernel density estimation has a slightly better performance. Therefore, it will appear in the final comparison.

NB handling of continuous variables	AUC
Kernel density estimation	0.7616
Categorization	0.75
Discrete values	0.7554

Table 4.7: AUC scores of the Naive Bayes classifiers

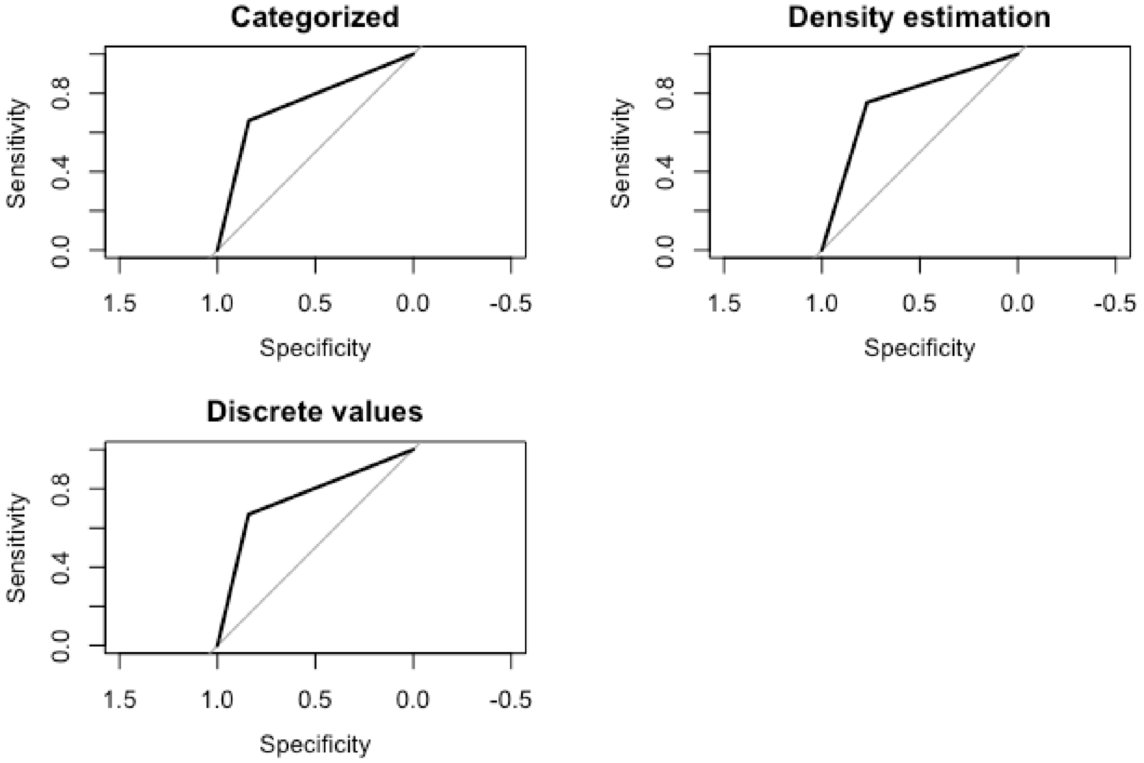

Figure 4.6: ROC curves of the Naive Bayes classifiers

4.4. Feedforward neural network

Training a proper feedforward neural network model required more preparation than other learning algorithms. First of all, the input data needed to be transformed. Categoric variables had to be transformed into dummy variables, like for the logistic regression. Furthermore, the numeric variables had to be normalised and brought into the range [0,1]. In order to achieve that, min-max normalisation was performed. It is defined as:

$$x' = \frac{x - min(x)}{max(x) - min(x)} \tag{4.5.1}$$

Afterwards, a set of hyperparameters needed to be tuned. The simplest and most efficient way to do that is to perform a grid search over all of the hyperparameters and to see which combination gives the best prediction power to the network. The parameters that were tested and compared were:

- The amount of gradient descent iterations/steps: 50, 100, 250, 500, 1000
- The amount of nodes in the single hidden layer: 3, 5, 10, 15, 20, 30, 40, 50
- Learning rate: 0.0001, 0.001, 0.01, 0.1
- Weight decay: 0.01, 0.1, 1

Different values for the amounts of gradient descent interactions and nodes in the hidden layer were chosen arbitrarily. The values for the learning rate were chosen according to a simple suggestion by Mitchell (1997), that it should be as low as possible. Weight decay values were chosen in accordance with Ripley and Venables (2004). The values for weight initialisation in the network were chosen as suggested by Marshald (2009) in the range between $-\frac{1}{\sqrt{n}}$ and $\frac{1}{\sqrt{n}}$, where n is the amount of input and hidden nodes in the network. Since different amounts of nodes in the hidden layer were tested, the average value was chosen, which means that the values for the initial weights were chosen from a range [-0.15, 0.15], and they followed uniform distribution. Lastly, the tangent hyperbolic function was the chosen activation function.
As on can see, there were 5x8x4x3 combinations of parameters to test. One way to accomplish that was to perform a simple exhaustive search, where all combinations are tested against each other. However, it would require a lot of computational time. According to Bergstra and Bengio (2012), random search is another effective solution, that allows to find models, that are as good as models configured by a pure full grid search, within a much smaller computational time. The random search can be optimized even more by defining early stopping criteria for the search. For this analysis, the early stopping criteria was to stop searching if the cross-entropy error has improved over the moving average of the best five models by less than 0.01. The results of the random grid search are shown in the Table 4.8:

Model	Iterations	Hidden	Learning rate	Weight decay	AUC	Cross-entropy
1	500	5	0.001	0.01	0.861	0.4026
2	250	5	0.1	1	0.862	0.4020
3	1000	5	0.01	0.1	0.863	0.3997
4	250	10	0.1	0.01	0.865	0.4027
5	100	30	0.1	0.1	0.866	0.3947

Table 4.8: Results of the random search for the hyperparameters

The random search was stopped after the first five iterations, since the algorithm did not observe a sufficient improvement in the cross-entropy error. The AUC score and the cross-entropy error were calculated on the validation set. The results show, that the differences in the five models are marginal. In order to balance out model complexity and predictive power, the model with 250 iterations, 10 hidden layers, learning rate of 0.1 and weight decay of 0.01 was chosen for the further comparison. The difference in the AUC to the „best“ model with the highest AUC is negligible, as well as in the cross-entropy error.

4.5. Final comparison

After the best models of all four supervised algorithms for a given dataset were defined, a final comparison on a test set of data could be carried out. The results are shown in Table 4.9 and Figure 4.7. Results show that in this analysis CART and Naive Bayes algorithms were outperformed by the logistic regression and feedforward artificial neural network. The latter two have almost the same AUC score, with logistic regression having a slight edge.

Supervised learning algorithm	AUC
CART	0.7097
Logistic Regression	0.7851
Naive Bayes	0.7636
ANN	0.7821

Table 4.9: Final comparison

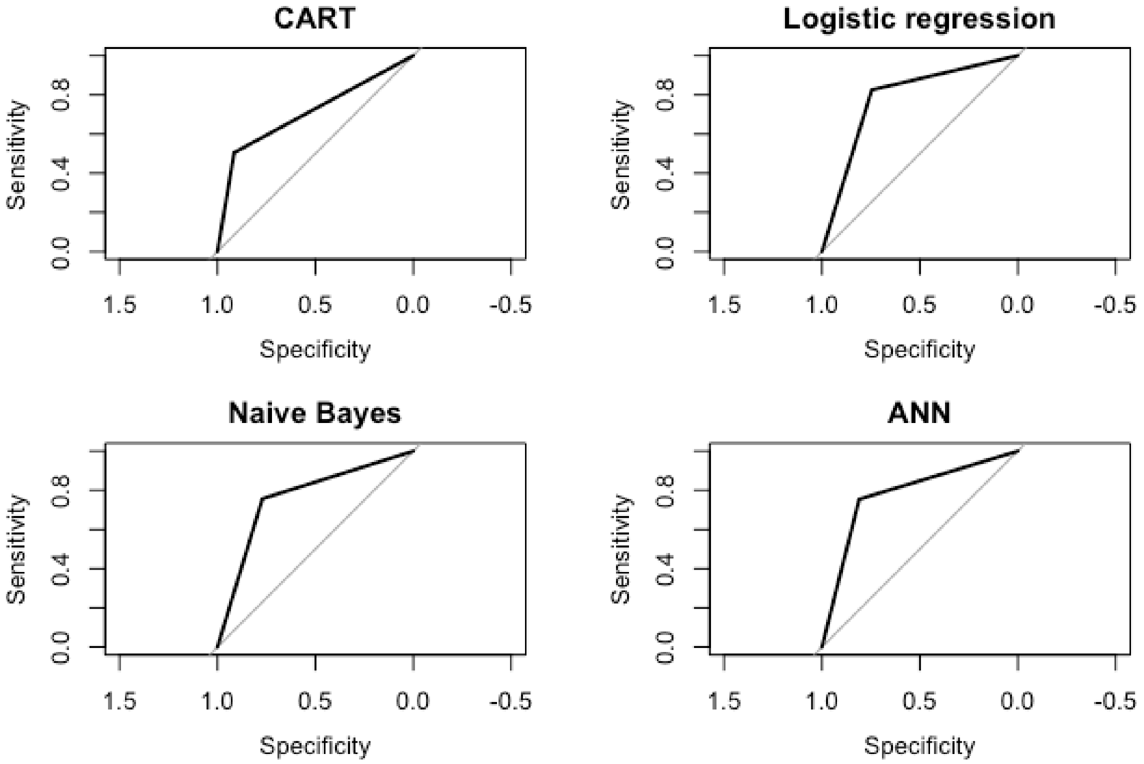

Figure 4.7: ROC curves for the final comparison

5 Discussion

By no means does this thesis represent an exhaustive empirical comparison of the four algorithms described before. Ideally, at least a couple more other supervised learning algorithms should have also considered in the analysis. There was simply not enough time to test out all the possible models that could be created for those supervised methods. Nevertheless, the results of the comparison in this thesis are congruent with a general consensus about the four algorithms analysed. The comparison has shown that different learning algorithms deliver significantly different results within the same classification problem.

When it comes to predicting binary outcomes, logistic regression or neural networks are first that come to mind, depending on the complexity of the task. Generally, they produce better predictions. This can also be observed in this thesis, since they both significantly outperform Naive Bayes and CART based on Area Under Curve measure.

Nevertheless, it does not mean that Naive Bayes or CART should be discarded, when it comes to building a predictive model. If one needs a white-box model, where the decision rules need to be retraced, decision tree is still one of the best options. Granted, the prediction errors will probably be higher, especially if the tree size needs to be reduced to increase comprehension. Naive Bayes is still often used, when it comes to e.g. natural language processing (Sang-Bum et al., 2006) thanks to its simplicity and robustness.

However, when it comes to predictions that have to be as precise as possible, ANN and logistic regression are more preferred. Areas that have such requirements are e.g. healthcare research and pharmaceuticals, and these areas use those algorithms for quite a while (Eftekhar et al., 2005). A study by Eftekhar et al., 2005 shows that in many cases artificial neural networks outperform logistic regressions. It makes sense, since neural networks are extremely flexible and if there is enough time for proper tuning of all of the hyperparameters, they most probably deliver best results. However, artificial neural networks lack transparency in decision making. Logistic regression offers both good predictive power and the ability to somewhat explain what happens inside the model.

To sum up, a perfect supervised learning algorithm for solving a binary classification problem does not exist. Every algorithm has its up- and downsides, there are always trade-offs. It depends on the goals of an analyst or a researcher, which algorithm will be used. Only when it comes to having solely best predictions possible, without the need to explain exactly why those predictions were made and there is enough time to develop it, one can say that artificial neural network is most certainly the best choice.

6 References

Andriyashin A., Härdle K. W., Timofeev R. (2008). *Recursive Portfolio Selection with Decision Trees.* SFB 649 Discussion Paper 2008-2009, pp. 5-14.

Bergstra J., Bengio Y. (2012). *Random search for Hyper-Parameter Optimization.* Journal of Machine Learning Research 13, pp. 281-305.

Breiman L., Friedman H. J., Olshen A. R., Stone J. C. (1984). *Classification and regression trees.* Chapman & Hall, pp. 27-103.

Breiman L. (1996). T*echnical Note: Some Properties of Splitting Criteria.* Machine Learning, Vol. 24, pp. 41-47.

Eftekhar B., Mohammad K., Ardebili E. H., Ghodsi M., Ketabchi E. (2005). *Comparison of artificial neural network and logistic regression models for prediction of mortality in head trauma based on initial clinical data.* BMC Med Inform Decis Mak, v. 5, https://dx.doi.org/10.1186/1472-6947-5-3.

Fawcett T. (2006). *An introduction to ROC analysis.* Pattern Recognition Letters 27, pp. 861-874.

Hosmer W. D., Lemeshow S. Jr., Sturdivant X. R. (2013). *Applied Logistic Regression, Third Edition,* pp. 1-61.

Klinke S. (2018), Slides from the lecture *Data Analysis II.*

Marshald S. (2009). *Machine learning: an algorithmic perspective*. Chapman & Hall/CRC, p. 58.

Mitchell M. T. (1997). *Machine learning*. McGraw-Hill, pp. 81–116.

Ng Y. A., Jordan I. M. (2001). *On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes*. Neural Information Processing Systems 2001, p. 2.

Ripley B. D., Venables W.N. (2002). *Modern applied statistics with S, fourth edition.* Springer, pp. 245-246.

Sang-Bum K., Kyoung-Soo H., Hae-Ching R., Sung H. M. (2006). *Some Effective Techniques for Naive Bayes Text Classification.* IEEE Transactions on knowledge and data engineering, vol. 18, no. 11, p. 1457.

The Economist (2017). *Machine-learning promises to shake up large swathes of finance.* https://www.economist.com/finance-and-economics/2017/05/25/machine-learning-promises-to-shake-up-large-swathes-of-finance

U.S. Census Bureau (2018). *About PUMS.* https://www.census.gov/programs-surveys/acs/technical-documentation/pums/about.html

U.S. Census Bureau (2018). *About American Community Survey.* https://www.census.gov/programs-surveys/acs/about.html

U.S. Department of Labor (2011). *The Fair Labor Standards Act of 1938*. https://www.dol.gov/whd/regs/statutes/FairLaborStandAct.pdf

7 Appendix

Coded variable name	Description	Variable type/variable values
AGEP	Age	Numeric; 1-99
COW	Class Of Worker	Categorical; NA (10) -Not in labor force who last worked more than five years ago or never worked, 1 - Employee of a private for-profit company or of an individual, 2 - Employee of a private not-for-profit organization, 3 - Local government employee, 4 - State government employee, 5 - Federal government employee, 6 - Self-employed in own not incorporated business, 7 - Self-employed in own not incorporated business, 8 - Working without pay in family business or farm, 9 - Unemployed and last worked five years ago or earlier or never worked
MAR	Marital status	Categorical; 1 - Married, 2 - Not married
school	Educational attainment	Categorical; 1 - below HS diploma, 2 - HS diploma or similar, 3 - College dropout, 4 - Associate's degree, 5 - Bachelor's degree, 6 - Master's degree, 7 - Professional degree beyond a bachelor's degree, 8 - Doctorate degree
SEX	Sex	Categorical; 1 - male, 2 - female
WKHP	Usual hours worked per week past 12 months	Numeric; 0 - did not work, 1-98 – 1 to 98 usual hours, 99 - 99 or more usual hours

Coded variable name	Description	Variable type/variable values
DIS	Disability	Categorical; 1 - With a disability, 2 - Without a disability
income	Total person's income in the past 12 months	Categorical; <50k - less than 50 thousand dollars, >=50k - 50 thousand dollars or more
RACWHT	White race recode (White alone or in combination with other races)	Categorical; 0 - No, 1 - Yes
WAOB	World area of birth	Categorical; 1 - US State, 2 - PR and US island areas, 3 - Latin America, 4 - Asia, 5 - Europe, 6 - Africa, 7 - Northern America, 8 - Oceania and at Sea

Declaration of authorship

I, Ansar Aynetdinov, hereby declare that I have not previously submitted the present work for other examinations. I wrote this work independently. All sources, including sources from the Internet, that I have reproduced in either an unaltered or modified form (particularly sources for texts, graphs, tables and images), have been acknowledged by me as such.
I understand that violations of these principles will result in proceedings regarding deception or attempted deception.

Ansar Aynetdinov

Berlin, August 19th, 2018